IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 15 June 2022, accepted 7 July 2022, date of publication 13 July 2022, date of current version 20 July 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3190632

RESEARCH ARTICLE

Detecting Latent Topics and Trends in Software Engineering Research Since 1980 Using Probabilistic Topic Modeling

FATIH GURCAN[1], GONCA GOKCE MENEKSE DALVEREN[2], NERGIZ ERCIL CAGILTAY[2], AND AHMET SOYLU[3]

[1]Department of Computer Engineering, Faculty of Engineering, Karadeniz Technical University, 61080 Trabzon, Turkey
[2]Department of Software Engineering, Faculty of Engineering, Atilim University, 06830 Ankara, Turkey
[3]Department of Computer Science, Norwegian University of Science and Technology, 2815 Gjøvik, Norway

Corresponding author: Ahmet Soylu (ahmet.soylu@ntnu.no)

ABSTRACT The landscape of software engineering research has changed significantly from one year to the next in line with industrial needs and trends. Therefore, today's research literature on software engineering has a rich and multidisciplinary content that includes a large number of studies; however, not many of them demonstrate a holistic view of the field. From this perspective, this study aimed to reveal a holistic view that reflects topics, trends, and trajectories in software engineering research by analyzing the majority of domain-specific articles published over the last 40 years. This study first presents an objective and systematic method for corpus creation through major publication sources in the field. A corpus was then created using this method, which includes 44 domain-specific conferences and journals and 57,174 articles published between 1980 and 2019. Next, this corpus was analyzed using an automated text-mining methodology based on a probabilistic topic-modeling approach. As a result of this analysis, 24 main topics were found. In addition, topical trends in the field were revealed. Finally, three main developmental stages of the field were identified as: the programming age, the software development age, and the software optimization age.

INDEX TERMS Corpus creation, research trends and topics, software engineering, text mining, topic model.

I. INTRODUCTION

The societal and economic impact of software is unquestionable. Accordingly, software engineering (SE) is emerging as a young and rapidly developing field. Today, the industrial sector continues to integrate software technologies and tools into their operational processes, which increases the demand for cutting-edge technology and innovative software applications capable of solving complex problems. In this context, the amount of research and applications in the field of SE are increasing exponentially. As a result, the current SE literature has a rich content that covers a wide spectrum [1] of research work containing emerging ideas about methods, technology, and practices [2], [3]. Hence, in order to provide a deeper understanding of the field, several studies have been conducted to analyze and assess the research landscape. The results of such analyses are very important for the improvement of SE education programs and for shedding light on how to better identify the trends in the SE field [4]. In addition, these analyses are essential for standardizing and better understanding the field of SE. Earlier studies have also reported the need for research on the syntheses of SE studies [5], [6], indicating the need for deeper analyses and a systematic approach to understanding the trends in the field. Currently, there is no systematic study covering the 40 years since the inception of the field. Previous studies either cover a limited time period [6]–[8] or present an overview of different databases [9]. Therefore, the active studies focusing on popular research topics in the field of SE need to be identified. The evolving trends of these research topics and their distribution can help to reveal the current status of the SE research community.

Accordingly, this study initially proposes a methodology for creating a corpus through the compilation of major

The associate editor coordinating the review of this manuscript and approving it for publication was Yang Liu.

publication sources in the field. The corpus was then created by considering studies published over the last 40 years. Additionally, SE research trends and topics were investigated by analyzing the studies in this corpus using a topic-modeling approach. The trends of these topics were further analyzed to provide some insights about their prospects in the near future. In addition, the developmental stages of the SE field were clarified and some suggestions for improving the SE classification system were provided.

II. BACKGROUND OF THE STUDY

The formulation of scientifically plausible novel research hypotheses requires a comprehensive analysis of the existing domain of specific knowledge [5]. Accordingly, review studies have been implemented by the SE community to provide meaningful evidence for various research and techniques. As detailed below, these studies can be classified into four groups.

A. REVIEWS ANALYZING THE MOST-CITED ARTICLES

The first group of studies consists of those analyzing the top articles in the field [6], [10]–[14]. Most of these were published in SE journals between 1999 and 2002 by Wohlin [10]–[12]. The aim of this series of articles was to present a list of the 20 most-cited studies based on the analysis and to invite the authors to contribute to a special section of the journal *Information and Software Technology* [11]. The articles were classified according to their research type via systematic mapping. The results of the study by Kitchenham [13] indicated that the articles were published in journals with an empirical content and that most were published in *IEEE Transactions on Software Engineering* (TSE). The study of Cai and Card focused on the most-cited articles on software metrics published between 2000 and 2005 [6]. They considered seven top SE journals and seven leading SE conferences. After analyzing 691 articles, they reported that 73% of the journal articles focused on 20% of the subjects in SE, including testing and debugging, management, and software/program verification, whereas 89% of conference articles focused on 20% of the SE subjects, including software/program verification, testing and debugging, and design tools and techniques [6]. Another study by Garousi and Fernandes [14] identified the articles in the field of SE that most influenced others based on the citation count and stated that identifying top-cited articles enabled researchers to observe the different types of presented and applied approaches and research methods. According to the results of this study, the top article was ''A metrics suite for object-oriented design'', with 1817 citations [14].

B. REVIEWS BASED ON BIBLIOMETRIC ANALYSIS

Another group of studies in the SE field were conducted as bibliometric analyses [15]–[23]. Among them, a series of bibliometric studies were conducted and presented in 12 articles by Wong *et al.* [15]–[17] that identified the top 15 scholars and institutions in systems and SE as an annual event between 1995 and 2006. The rankings were based on the number of articles in a selected set of top SE journals [15]–[17]. Another bibliometric study by Garousi and Varma reported the correlation analysis of the SE research productivity of Canadian provinces versus their national grant amounts [18]. Freitas and De Souza [19] focused on the sub-fields of SE, presenting a bibliometric study over ten years of the search base. Additionally, another bibliometric analysis by Farhoodi *et al.* stated that most active authors in this field were from the US [20]. Another example conducted by Garousi and Ruhe [21] was a bibliometric/geographic study comprising an assessment of 40 years of SE research between the years 1969 and 2009 using a set of 26,624 SE articles indexed by the ISI Web of Knowledge. These articles were examined to identify the most active countries in SE and according to the findings, 60% of the SE literature was contributed by 7% of all countries globally [21]. Fernandes reported a bibliometric study [23] which concentrated on authorship trends in SE by collecting and analyzing nearly 70,000 entries for 122 conferences and journals between the years of 1971 and 2012. Garousi published a bibliometric study [22] comprising an assessment of Turkish SE scholars and institutions covering the years 1992 to 2014, and according to the results, Turkey produced only about 0.49% of worldwide SE research.

C. SYSTEMATIC REVIEWS (SRs)

The third group of studies were conducted as systematic reviews (SRs) in the field of SE [1], [16], [24]–[27]. Most of these studies implemented the guidelines for undertaking SRs according to the medical standards introduced by Kitchenham [28]. These were then revised by Biolchini *et al.* [29] to consider the practical problems related to the usage of guidelines. Afterwards, Kitchenham and Charters [30] incorporated the approaches to SRs proposed by sociologists. Kitchenham *et al.* [31] claimed that meta-analyses could not be conducted without established sampling protocols for appropriately defined SE populations and a set of standard measures. Software engineers began to use SRs and many researchers in the field also started to comment on the SR process itself. Kitchenham and Brereton [1] presented one of the first studies which commented on problems related to the application of SRs. This was followed by a considerable increase in the number of SRs that have been conducted over the last 10 years [32]–[35]. For example, Hall *et al.* [5] conducted a systematic review on the motivations of software engineers by analyzing 92 studies published between 1980 and 2006. Garousi *et al.* aimed to determine the challenges of avoiding risks to the collaboration by being aware of these challenges. This led to an SR being conducted, and from the thematic analysis, 10 challenging themes and 17 best-practice themes were identified [24]. Another study by Garousi *et al.* involved the systematic mapping of secondary studies in software testing. The aim was to summarize indexes that facilitated finding the most relevant information from secondary studies supporting evidence-based

decision making in any given area of SE [25]. They used the systematic-mapping approach and included 101 secondary studies in the area of software testing between 1994 and 2015 [25]. According to the results, in terms of the number of secondary studies, the model-based approach was the most popular testing method, with web services being the most popular system under testing, and regression testing being the most popular testing phase [25]. A systematic mapping study was conducted by Rodríguez-Pérez *et al*. to identify the diversity in software engineering [36]. They found that the studies had a gender diversity perspective focusing on revealing gender bias or gender differences [36]. In order to observe the trends of empirical methods applied in SE, Zhang *et al.* conducted an empirical software engineering mapping study using 538 selected articles published from January 2013 to November 2017 [37]. Ramirez *et al.* performed a SR from a total of 669 articles, and identified 26 primary studies in which some open concerns were discussed as well as potential future trends for the research community [26]. To help improve software performance and quality, Nazar *et al.* conducted a literature review study summarizing software artifacts by focusing on bug reports, source codes, mailing lists, and developer discussions [38]. In another study, Kitchenham *et al.* analyzed 20 relevant studies and reported that the topic areas covered by SRs were limited [33].

The most common criticisms of SRs were that they required a great deal of time, that certain SE digital libraries were not appropriate for broad literature searches, and that assessing the quality of empirical studies of different types was difficult [1]. Although a valuable tool for SE, conducting SRs remains a time-consuming and difficult task [35], [39], [40]. Additionally, in order to provide a general picture for the relevant fields, it is not always possible to consider a great number of publications when conducting these reviews.

D. REVIEWS BASED ON TEXT MINING

In order to obtain a deeper understanding of the field, some studies applied text-mining techniques [3], [7]–[9], [41]–[43]. For example, the study by Cosh *et al.* [7] presented an automatic method for extracting and examining key research themes using natural language processing. This technique makes it possible to parse a large collection of articles, and has been applied to over 8,000 articles published in the SE field over the past 20 years. With the goal of identifying the structure of research articles in SE, another study by Mathew *et al.,* conducted using text mining, explored 35,391 SE articles from 34 leading SE venues over the last 25 years [8]. Garousi *et al.* used topic modeling to automatically generate the most probable topic distributions for thematic analysis in the SE field using text mining of article titles. They identified the hot research topics in SE as web services, mobile and cloud computing, industrial studies, and source-code and test generation [3]. The study by Barua *et al.* presented a methodology for analyzing the textual content of Stack Overflow discussions via the statistical topic-modeling technique of Latent Dirichlet Allocation (LDA) [9], which allowed the discovery of hot topics present in developer discussions.

Because of the high volume of SE studies, text-mining approaches provide an opportunity to better understand the field and to draw future support for decision makers and researchers in the field. However, analysis of earlier studies has revealed that they are mainly focused on bibliometric analysis, which provides very limited perspectives for the field. Even though there have been some attempts at text-mining-based analysis in SE studies, limitations exist, either in the corpus creation methods or in the analysis approaches to the created corpus. Additionally, the majority of these studies have considered a limited timeline for the field. As the method of corpus creation is a critical factor in the results, approaches that are more systematic are needed for automated text-mining studies. In order to provide a wider perspective of SE, this current study first proposed a systematic and objective approach for SE studies corpus creation. Following a deep analysis, several insights into the field of SE were provided. The details of the methodology are described below.

III. METHOD

The methodology of this study, which aimed to analyze the last forty years of SE research, is presented in three phases. First, the process of corpus creation is presented. The second phase consists of the data pre-processing operations. In the final phase, data analysis procedures and the LDA-based topic modeling process are given. The methodology of the study is outlined in Fig. 1.

A. CREATION OF CORPUS (CC)

The literature in the SE field can be considered very rich as it includes a high number of conferences and journals specific to the field. Hence, in such a review study, the selection process of these data sources and the published articles in them is critical [44], [45]. For this study, the following sequential procedures were employed to select data sources and create the corpus for experimental analysis:

CC1. To create the corpus of the study including the core conferences and the journals (publication sources) in the SE field, first, all publication sources addressing the keywords ''software'' and ''programming'' were selected from the Scopus database, which indexes all core SE conferences and journals [44]–[46]. Scopus is one of the largest abstract, citation, and bibliometric databases for peer-reviewed academic literature including scientific journals, books, and conference proceedings [46]. Therefore, it was used for the creation of the corpus of this study.

CC2. From the search results conducted in CC1, a corpus data source list (CDSL) was created including the names of all selected publication sources.

CC3. The H5-index (H5-I) values of the selected publication sources were obtained from the Google Scholar Metrics and the CDSL was ordered according to the H5-I values,

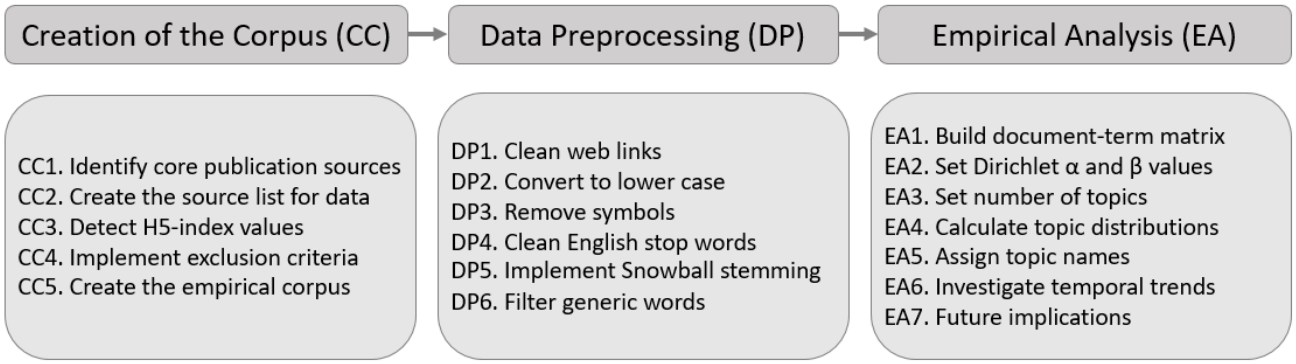

FIGURE 1. Overview of the methodology of the study.

which define the h-index for articles published in the last five years completed. The ''h'' stands for the largest number and thus, each of the ''h'' articles published between the years 2015 and 2019 has at least ''h'' citations.

CC4. In order to create a core collection for the SE field, we decided to exclude the conferences and publication sources that had not yet reached academic maturity. For this reason, we used the H5-I values. Accordingly, after an analysis of the all publication sources in the field, in order to set an objective threshold, the researchers set the H5-I value in this study as 20. Although by changing this H5-I value, a corpus with some variations can be created via exclusion or inclusion of different publication sources, the current study considered only the publication sources satisfying this requirement. In other words, with the aim of creating a core collection specific to the SE field, the publication sources that had achieved academic maturity and the citation score (an H5-I value of greater than or equal to 20) were identified. Accordingly, this final version of the CDSL included 28 conferences and 16 journals (44 publication sources) covering up to June 20, 2020.

CC5. In this step, the articles in English in the publication sources listed in the CDSL between 1980 and 2019 were identified. Next, the ''title'', ''abstract'', and ''author keywords'' sections of each article were downloaded from the Scopus database on June 20, 2020. Because the year 2020 had not been completed at that time, the publications of 2020 were not included in the corpus. The ''title'', ''abstract'', and ''author keywords'' describe the content of each study in a condensed form. Therefore, for the corpus, these data were considered for analysis by eliminating the repetitions and focusing on the core of each article. Finally, 57,174 articles were included in the corpus of this study. The number of these articles (N) in five-year periods and their percentage (%) of the total number of articles for that period are given in Table 1.

The number of articles (N) in the field published during the last 20 years (80%) was four times higher than those published during the first 20 years (20%), indicating an increasing interest in the field of SE. The H5-I values of all the data sources, the number of articles (N) and their percentage (%) published by these data sources are given in Table 2 (ranked by N). The majority of the articles were published in *ACM/IEEE International Conference on Software Engineering* (10.44%), followed by *Journal of Systems and Software* (7.56%), and *IEEE Transactions on Software Engineering* (5.57%).

TABLE 1. Distribution of articles in five-year periods.

Periods	N	%
1980-1984	1,515	2.65
1985-1989	2,556	4.47
1990-1994	3,617	6.33
1995-1999	4,852	8.49
2000-2004	6,148	10.75
2005-2009	10,502	18.37
2010-2014	12,338	21.58
2015-2019	15,646	27.37

B. DATA PREPROCESSING (DP)

Before implementation of LDA, the following textual data preprocessing steps were undertaken to prepare the corpus in a suitable format for probabilistic topic modeling [47], [48].

DP1. All web links in the dataset were deleted.

DP2. All text was converted to lower case and word tokenization was employed.

DP3. Punctuation, numbers, symbols, links, and formatting were deleted.

DP4. High-frequency English stop words not contributing to the creation of topics with semantic integrity, such as ''and'', ''for'', ''an'', ''the'', and ''is'' were removed [44]–[47].

DP5. A snowball-stemming algorithm was applied to the textual corpus to remove the words from their derived suffixes and reduce them to the root form (e.g., the words ''testing'', ''tester'', and ''tested'' all combine into ''test'') [49].

DP6. Generic words frequently used in academic work that do not contribute to semantics, such as ''article'', ''paper'', ''research'', ''study'', and ''copyright'', were filtered,

TABLE 2. Publication sources included in the corpus.

No	Type	Publication Name	H5-I	N	%
1	C	ACM/IEEE International Conference on Software Engineering	75	5,967	10.44
2	J	Journal Of Systems And Software	52	4,320	7.56
3	J	IEEE Transactions On Software Engineering	48	3,185	5.57
4	C	International Colloquium on Automata, Languages and Programming (ICALP)	32	3,138	5.49
5	J	IEEE Software	41	3,116	5.45
6	J	Information And Software Technology	55	2,964	5.18
7	J	Software Practice And Experience	27	2,902	5.08
8	J	Science of Computer Programming	33	1,885	3.30
9	C	ACM SIGPLAN International Conference on Object-Oriented Programming, Systems, Lang… (OOPSLA)	34	1,818	3.18
10	C	IEEE/ACM International Conference on Automated Software Engineering (ASE)	40	1,705	2.98
11	C	International Symposium on Software Reliability Engineering (ISSRE)	25	1,644	2.88
12	C	IEEE International Conference on Software Testing, Verification and Validation (ICST)	27	1,408	2.46
13	C	IEEE International Conference on Web Services	21	1,387	2.43
14	C	ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL)	51	1,384	2.42
15	C	ACM SIGSOFT International Symposium on Foundations of Software Engineering	51	1,220	2.13
16	C	ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI)	51	1,213	2.12
17	C	International Conference on Principles and Practice of Constraint Programming	22	1,206	2.11
18	C	European Conference on Object-oriented Programming (ECOOP)	21	1,093	1.91
19	C	IEEE International Requirements Engineering Conference	25	1,054	1.84
20	J	International Journal of Parallel Programming	23	925	1.62
21	C	International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS)	39	891	1.56
22	C	ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming (PPOPP)	31	854	1.49
23	J	Empirical Software Engineering	49	833	1.46
24	C	IEEE Annual Computer Software and Applications Conference (COMPSAC)	24	791	1.38
25	C	ACM SIGPLAN International Conference on Functional Programming (ICFP)	27	754	1.32
26	C	International Symposium on Empirical Software Engineering and Measurement, ESEM	20	753	1.32
27	C	International Symposium on Software Testing and Analysis	35	735	1.29
28	J	International Journal on Software Tools for Technology Transfer	27	701	1.23
29	C	European Symposium on Programming (ESOP)	26	694	1.21
30	J	Software And Systems Modeling	38	648	1.13
31	J	Software Quality Journal	22	631	1.10
32	J	Theory and Practice of Logic Programming	24	615	1.08
33	C	Mining Software Repositories	38	523	0.91
34	C	IEEE International Conference on Program Comprehension	25	506	0.89
35	C	IEEE International Symposium on Performance Analysis of Systems and Software	22	485	0.85
36	J	Journal of Software Evolution and Process	22	471	0.82
37	J	ACM Transactions on Software Engineering and Methodology (TOSEM)	28	467	0.82
38	C	IEEE International Conference on Software Maintenance and Evolution	33	418	0.73
39	J	Requirements Engineering	25	392	0.69
40	J	Software Testing Verification And Reliability	20	379	0.66
41	C	International Conference on Software Analysis, Evolution, and Reengineering (SANER)	30	329	0.58
42	C	International Conference on Evaluation and Assessment in Software Engineering	21	314	0.55
43	C	International Conference on Agile Processes in Software Engineering and Extreme Programming (XP)	20	260	0.45
44	C	International Symposium on Software Engineering for Adaptive and Self-Managing Systems	24	196	0.34
C: Conference, **J**: Journal			**Total**	57,174	100

considering their contribution to the semantic integrity of the topics in a way similar to DP4. Although such words and their derivatives are frequently mentioned in academic articles, they do not contribute semantically to the formation of the main topics reflecting the field of SE [48].

These preprocessing steps implemented at this stage are accepted processes that have been effectively applied extensively for the topic modeling of scientific texts in many previous studies [9]–[44], [48]–[50]. Therefore, the preprocessing steps were adapted to this study in accordance with the topic-modeling systematic suggested in the literature [9], [44], [48], [50], [51].

C. EMPIRICAL ANALYSIS (EA)

Topic modeling is an effective information extraction technique in text-mining studies performed on large amounts

of documents. Through a probabilistic and iterative process, it allows the automatic exploration of key topics from a text collection comprised of a huge amount of documents, without the need for tags, training data, or predefined taxonomies [52].

Numerous topic-modeling algorithms with similar perspectives are presented for semantic analysis of scientific texts including Correlated Topic Model (CTM), Dynamic Topic Model (DTM), Dirichlet Multinomial Regression (DMR), Non-Negative Matrix Factorization (NMF), Hierarchical Dirichlet Process (HDP), Hierarchical Latent Dirichlet Allocation (HLDA), Labeled Latent Dirichlet Allocation (LLDA), and Latent Dirichlet Allocation (LDA) [48]–[53]. The consistency calculation approach for estimating the optimal number of topics is quite limited for the majority of these algorithms (LLDA, NMF, CTM, and DMR) [48]–[53]. Although the HDP and HLDA algorithms offer an approach that automatically identifies the optimal number of topics, they require many prior parameters that need fine-tuning to achieve this process [48]–[54]. Therefore, the HLDA and HDP algorithms did not reveal acceptable results for this study in terms of semantic consistency of the topics. Among all the topic-modeling approaches, LDA is the most widely used algorithm that forms the basic background of topic modeling [48]–[53]. In addition, the LDA model provides many systematic methods and tools for revealing the optimal topics and their coherence values [48]–[53]. For these reasons, in this study, the LDA [52] algorithm was chosen in order to achieve a topic-modeling analysis that would reveal the themes and trends of SE.

Because LDA is a probabilistic and generative topic-modeling technique, topics identified using LDA are represented as probability distribution of words in the corpus. From this perspective, topics are created with word groups that tend to coexist a great deal within the documents making up the collection. In this respect, the words that make up the topics identified using LDA are closely related semantically, which allows each of these word groups to form a meaningful topic as a whole [55].

In the experimental analysis phase of this study, the fitting and implementation of LDA topic modeling on the SE corpus included several tasks and procedures. These sequential operations were performed by using the tmtoolkit [47], which includes the set of tools developed with Python for text mining and topic modeling described as follows:

EA1. First, the textual corpus was converted into a numerical matrix format referred to as the document-term matrix (DTM), in which each row represents an article and each column represents a unique word in the corpus in line with the ''bag of words'' approach [55]–[57].

EA2. The Dirichlet parameters α (the prior of the per-document topic distributions) and β (the prior of the per-topic word distributions) were estimated on this matrix [52]. In this context, the LDA model was fitted with $\alpha = 0.1$ and $\beta = 0.01$ values, which have been recommended in the literature for the topic modeling of short texts [58].

EA3. The LDA topic-modeling algorithm fitted with these parameters was employed with different numbers of topics (K) varying between 10 and 64 in order to empirically select the ideal number of topics. During this process, with the aim of evaluating the semantic integrity and consistency of the topics identified by LDA, the coherence metric C_V [59] was calculated for each topic model adapted from the 10 to 64 topics. The clarity and semantic consistency of the identified topics for each model of K were evaluated considering the C_V scores. Within these C_V scores [59], a maximum coherence score ($C_V = -1.9651287$) revealing the ideal topic-word distributions and semantic consistency for the topics was obtained with the topic number of $K = 24$. Therefore, 24 topics were used in the subsequent analysis.

EA4. The distribution percentage of each identified topic was calculated using the first 10 words describing each topic with the highest frequency.

EA5. Topic labels were assigned manually by evaluating all the words of each topic and at the same time, considering the high-frequency ones that ranked higher.

EA6. At this stage, the temporal trends of each identified topic were defined. The trend analysis was conducted for each of the five-year periods. The developmental stages of the SE field in each period were revealed through the interpretation and evolutions of these temporal trends.

In the literature, temporal trend analysis of each topic has mainly been conducted descriptively. In this study, we also analyzed the increasing rates of each topic by considering their acceleration values in five year periods (AC = average acceleration in five-year periods). Additionally, Y% (percentage of the topics in the same yearly period) and C% (percentage of the topics in the corpus) were also calculated. The calculation details are given step-by-step in Appendix A. Thus, the aim was to provide a wider perspective in order to get a deeper understanding of future topical trends. Hence, the acceleration analysis delivered a perspective on the increasing or decreasing trends of the topics. Furthermore, in order to better understand the developmental stages of the topics, the top five topics, depending on the mean number of articles published in each period, were also analyzed. This analysis provided a figure that gave a better understanding of the impact of each topic during each period. Based on this figure, the developmental stages of the SE were also identified.

EA7. The last stage revealed trend-lines indicating the direction of the temporal trends of each identified topic, and then, the implications regarding the near future movements of these topics were graphically illustrated.

IV. RESULTS

First, this section gives the descriptive results of the study demonstrating the distribution of the articles by years, subject areas, and publication sources. The topics generated by LDA are then presented and analyzed.

TABLE 3. Topics and top keywords revealed by LDA.

Topic Name	Keywords	%
Empirical Software Engineering	softwar*; engin*; experi*; develop*; practic*; empir*; discuss*; industri*; student*; communiti*	6.62
Project	softwar*; develop*; project*; process*; manag*; product*; team*; agil*; practic*; improv*	6.43
Architecture	system*; softwar*; architectur*; compon*; design*; adapt*; develop*; applic*; configur*; environ*	5.74
Logic Programming	logic*; program*; semant*; languag*; function*; express*; theori*; oper*; proof*; set*	5.44
Software Metrics	softwar*; model*; predict*; measur*; metric*; estim*; qualiti*; reliabl*; data*; defect*	4.99
Programming Languages	program*; languag*; compil*; code*; implement*; generat*; transform*; function*; describ*; programm*	4.77
Parallel Computing	perform*; parallel*; comput*; schedul*; memori*; applic*; optim*; time*; processor*; algorithm*	4.74
Specification and Verification	model*; specif*; verif*; formal*; system*; check*; properti*; verifi*; time*; behavior*	4.68
Source Code	code*; sourc*; chang*; softwar*; develop*; open*; refactor*; project*; api*; evolut*	4.67
Program Analysis	analysi*; program*; static*; dynam*; execut*; techniqu*; flow*; depend*; path*; data*	4.31
Process Modeling	model*; process*; product*; transform*; develop*; uml*; languag*; featur*; line*; domain*	4.29
Design Tools	design*; tool*; user*; pattern*; develop*; interact*; system*; visual*; interfac*; support*	4.29
Algorithm	algorithm*; graph*; time*; bound*; tree*; approxim*; complex*; comput*; set*; random*	4.16
Testing	test*; generat*; coverag*; techniqu*; suit*; autom*; mutat*; effect*; softwar*; input*	4.11
Requirements	requir*; system*; analysi*; engin*; specif*; process*; qualiti*; goal*; inform*; traceabl*	3.86
Object-Oriented	object*; class*; object-ori*; java*; program*; languag*; implement*; design*; system*; librari*	3.85
Constraint Optimization	constraint*; algorithm*; optim*; search*; solut*; solv*; variabl*; solver*; effici*; strategi*	3.77
Data	data*; databas*; queri*; learn*; inform*; search*; cluster*; mine*; retriev*; structur*	3.58
Distributed Systems	distribut*; network*; communic*; system*; protocol*; process*; messag*; agent*; node*; comput*	3.25
Web Services	servic*; web*; applic*; cloud*; composit*; resourc*; user*; workflow*; comput*; framework*	3.10
Fault Detection	bug*; fault*; detect*; error*; failur*; report*; techniqu*; debug*; local*; fix*	2.97
Concurrency	concurr*; memori*; program*; transact*; thread*; data*; synchron*; implement*; execut*; lock*	2.43
Mobile	mobil*; app*; devic*; applic*; android*; user*; energi*; develop*; game*; platform*	2.08
Security	secur*; vulner*; polici*; attack*; scheme*; access*; privaci*; inform*; protect*; imag*	1.88

A. TOPIC MODELING ANALYSIS

After applying the LDA analyses to the selected publications, 24 topics (Table 3) were revealed together with the top ten keywords for each topic. The topic names were given mainly by considering the top three keywords, except for the topics ''Software Metrics'' and ''Empirical Software Engineering'', which were named by considering all keywords in general. As seen in Table 3, the highest ratio for the analyzed topics was for ''Empirical Software Engineering'' (6.62%), followed by the topics ''Project'' (6.43%) and ''Architecture'' (5.74%).

The topics ''Security'' (1.88%) and ''Mobile'' (2.08%) had the lowest ratios. In order to gain a deeper understanding of the studies conducted on each topic, their trend-lines, developmental analysis, volume, and acceleration were calculated, and the results are given in the next section.

B. TREND-LINE ANALYSIS

With the aim of better understanding the temporal trend-lines of the topics identified and their development stages over the last 40 years, the percentage and acceleration of each topic were analyzed for each five-year period. In this context, the percentage of each topic in the corpus (C%) and the percentage of the topics normalized by years (Y%) are specified in the table presented in Appendix-A. The average acceleration (AC) value of the topics for each five-year period is also given in this table. Detailed descriptions for the calculations of these metrics are also given in Appendix-A. In order to visualize these accelerations, the acceleration graphs of all topics are presented in Fig. 2. Here, the blue lines show the average acceleration (AC) values of the topics for each period and the red lines represent the linear trend-lines estimating their movements in the near future. More specifically, red trend-lines above the x-axis indicate the topics with positive acceleration, e.g., ''Source Code'', ''Data'', and ''Mobile'', whereas those below the axis indicate negative acceleration, e.g., ''Programming Languages'', ''Logic Programming'', and ''Distributed Systems''.

C. OVERALL ACCELERATION ANALYSIS (LAST 40 YEARS)

In order to better understand the accelerating behavior of the topics, the positively and negatively accelerating topics from 1980 to 2019 are represented in Figs. 3 and 4. Fig. 3 indicates that sixteen topics were accelerating positively, whereas among them the topic ''Source Code'' is accelerating at a significantly higher rate (0.21) compared to the other topics. On the other hand, as seen in Fig. 4, eight topics were showing a negative acceleration values between 1980 and 2019, with the topic ''Programming Languages''

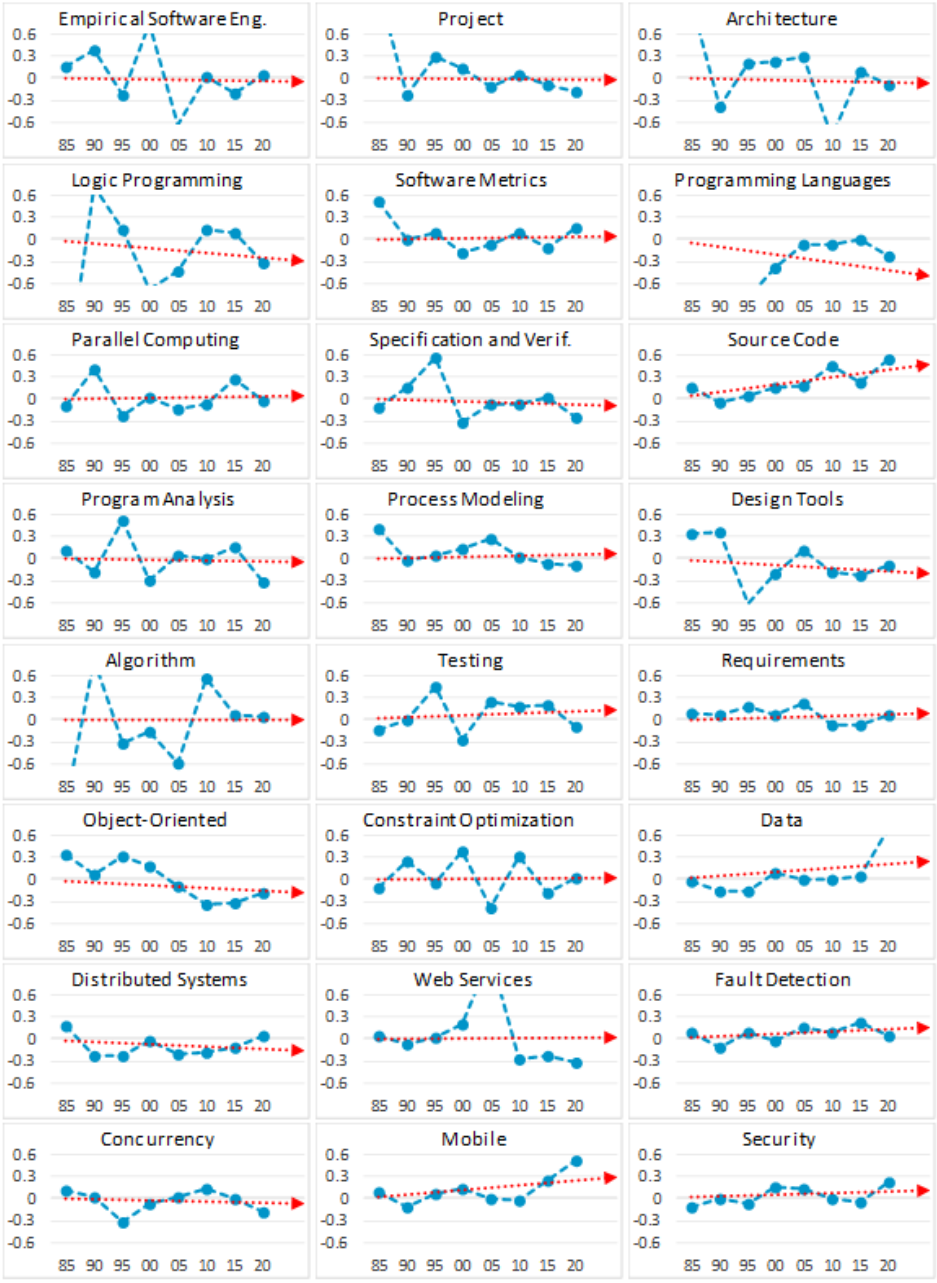

FIGURE 2. Acceleration graphs of the identified topics.

having the lowest acceleration value (−0.54) compared to the other topics (Fig. 4).

D. RECENT ACCELERATION ANALYSIS (LAST 5 YEARS)

To better understand the recent acceleration values and future directions, the acceleration values of the topics over the last five years (between 2015 and 2020) were also analyzed, as shown in the Figs. 5 and 6. Eleven topics exhibited a positive recent acceleration, whereas thirteen topics showed negative acceleration. Among the positively accelerating topics, "Data" had a significantly higher acceleration value (0.70) compared to the others (Fig. 5). Moreover, the topics "Source Code" (0.54), "Mobile" (0.53), "Security" (0.22), and "Software Metrics" (0.16) can also be considered as having higher recent acceleration rates. As seen from Fig. 6, the topics including "Logic Programming" (−0.33), "Web Services" (−0.31), and "Program Analysis" (−0.31) had the lowest recent acceleration values.

E. DEVELOPMENTAL ANALYSIS

In order to provide a deeper understanding of the developmental stages of the topics, the study further analyzed the top five topics according to the mean number of articles published

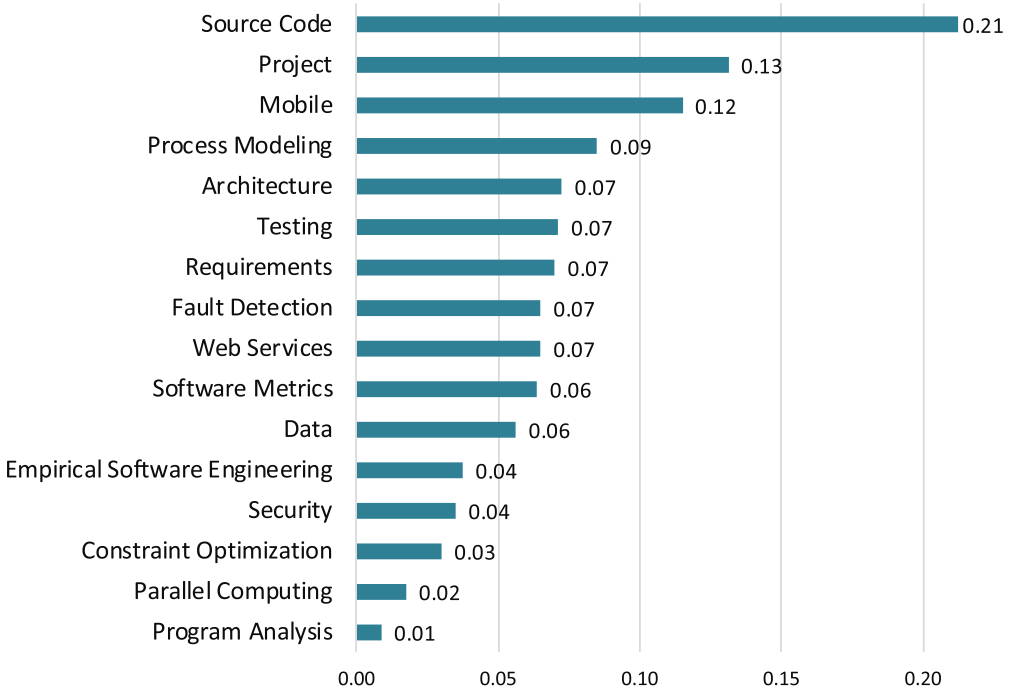

FIGURE 3. Positively accelerating topics from 1980 to 2019.

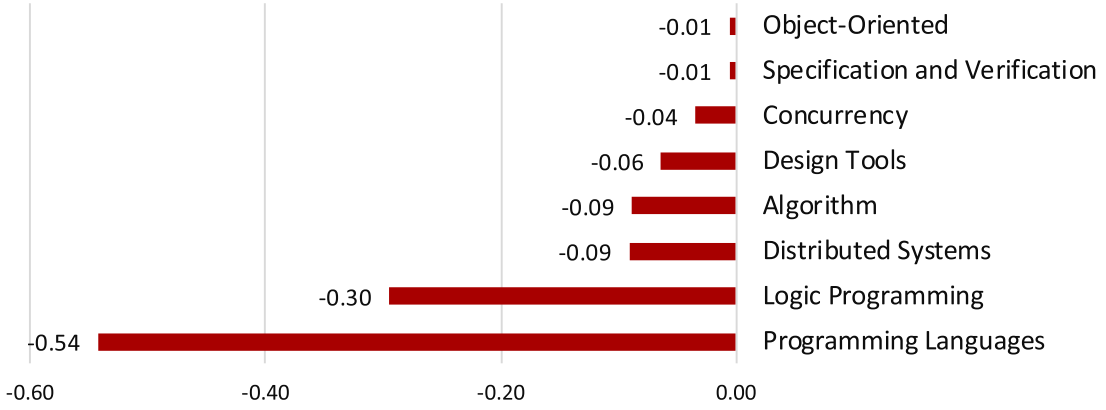

FIGURE 4. Negatively accelerating topics from 1980 to 2019.

in each period. These topics (from 1980 to 2020) are represented in random order as rectangles in Fig. 7. The top five topics from 1980 to 1985 were "Logic Programming", "Distributed Systems", "Algorithm", "Design Tools", and "Programming Languages". Furthermore, from 1980 to 1995, "Programming Languages", "Design Tools", "Logic Programming", and "Algorithm" were some of the dominating topics for SE studies. Based on this figure, the period between 1980 and 1995 can be named the "Programming Age" of the field during which programming concepts such as "Logic Programming", "Algorithm", "Parallel Computing", and "Programming Languages" were highly studied topics. However, between 1995 and 2010, studies on "Empirical Software Engineering", "Project", and "Architecture" began to dominate SE, indicating a strong focus on software development interests. During this period, studies on the software topics of "Specification and Verification", "Object-Oriented", "Web-Services", and "Process Modeling" were also especially studied. Accordingly, the period from 1995 to 2010 can be named the "Software Development Age". Finally, during the last decade, from 2010 to 2020, the studies on "Empirical Software Engineering", "Project", and "Architecture" continued to dominate the field, with research being focused on the software topics of "Testing" and "Source Code" and thus, this era can be referred to as the "Software Optimization Age" (Fig. 7).

V. DISCUSSION

The results of this current study show that there was an increased number of articles published in the SE field

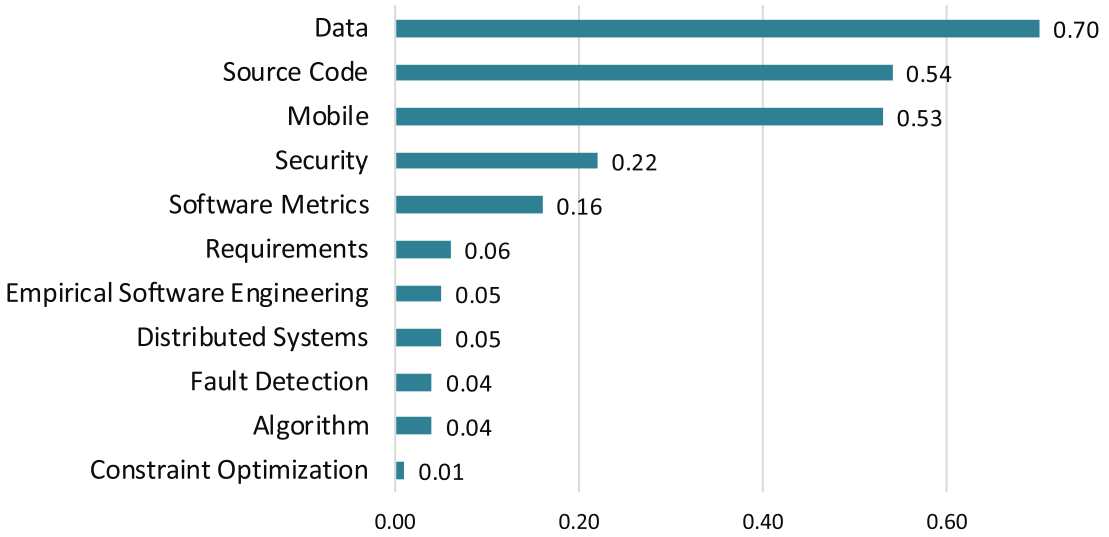

FIGURE 5. Topics having recent positive acceleration percentages from 2015 to 2019.

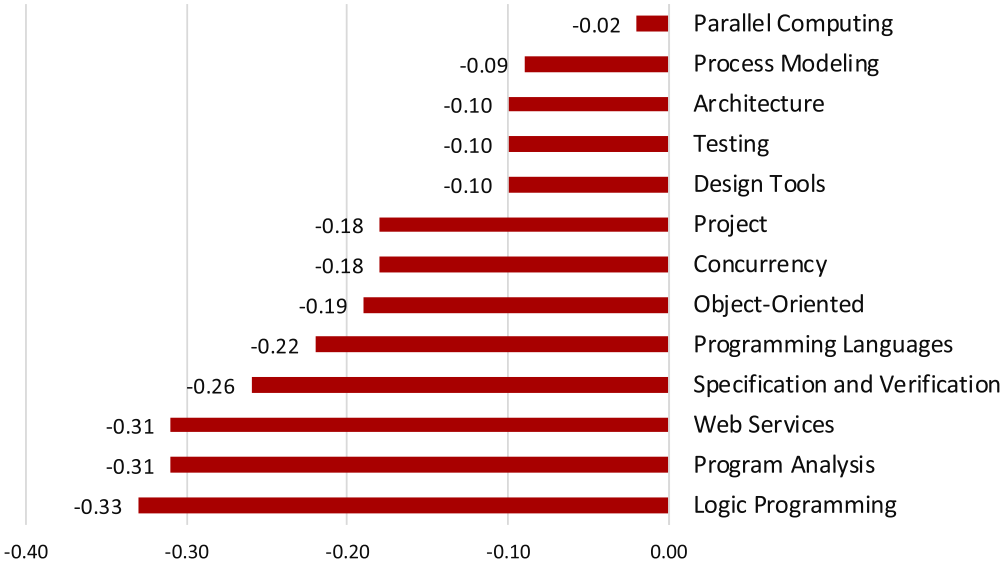

FIGURE 6. Topics having recent negative acceleration percentages from 2015 to 2019.

between 1980 and 2019, with slight decreases or increases in some years (Appendix-A). In their analysis of the period from 1969 to 2009, Garousi and Ruhe [21] also reported that the number of studies in the field of SE had accelerated after 1980, and they observed similar slight decreases and increases in the number of publications. In the current study, from 1980 to 2019, 44 top journals and conferences in the SE field had published around 3,000 studies a year. Garousi and Mäntylä [3] reported that there were between 6,000 and 7,000 articles published each year when all the studies having the "software" keyword were considered. These numbers are consistent according to the data source selection criteria of both studies and indicate an increasing interest in SE studies. The current study presents four main contributions to the field of SE, which are discussed below under the headings of Corpus Creation, SE Classification Systems, Developmental Stages of SE, and Topic Trends.

A. CORPUS CREATION

Cai and Card [6] conducted a study to analyze the research topics in the field of SE. They analyzed seven top SE journals that were all included in this current study (Table 2, No: 2, 3, 5, 6, 23, 37, and 40). Datta *et al.* [60] also considered nine of the same common data sources (Table 2, No: 2, 3, 5, 10, 11, 15, 18, 27, and 37) from 16 venues. In another study, Mathew *et al.* [8] analyzed 35,391 published articles from 34 conferences and journals in the SE field from 2009 to 2016. In their study, among the corpus of venues considering the selected conferences and journals, 22 of them (out of 34 publications) were common (Table 2, No: 1, 2,

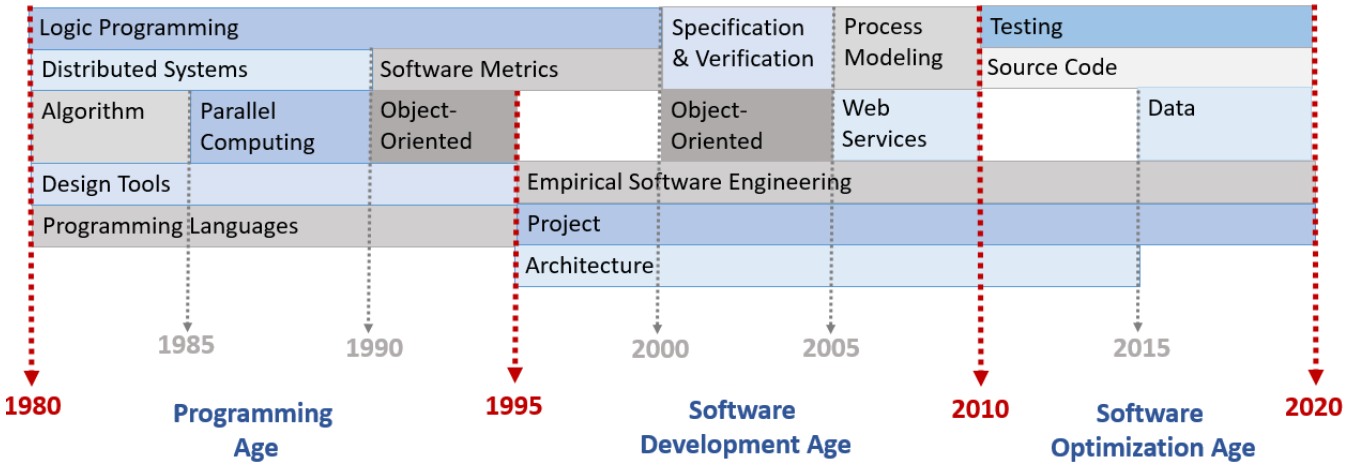

FIGURE 7. Developmental stages of SE studies.

3, 5, 6, 7, 10, 12, 15, 19, 23, 26, 27, 30, 31, 33, 34, 36, 37, 39, 40, and 41) compared with the current study (out of 44 publications). There were 22 publications (Table 2, No: 4, 8, 9, 11, 13, 14, 16, 17, 18, 20, 21, 22, 24, 25, 28, 29, 32, 35, 38, 42, 43, and 44) included in the current study according to our selection method that were not found in the 2018 study of Mathew *et al.* [8]. Even though Mathew *et al.* [8] and this current study both considered the H5-I values of the Google Scholar, the cut-off point of the H5-I was not clear in Mathew *et al.* [8]. Additionally, Mathew *et al.* [8] considered the H5-I values published in year 2012, whereas the current study considered those published in 2019, which could be the main reason for the differences in the corpus of the two studies.

The study conducted by Garousi and Mäntylä[3] used text-mining techniques. The authors performed a search by using the keyword "software". However, this choice of words might have created the risk of selecting studies that were not directly related to the field of SE. Additionally, this approach might also have created the risk of missing some related studies not having "software" as the keyword (e.g., "mobile applications"), but directly addressing the field of SE. Cosh *et al.* [7] also conducted research to identify themes and trends in the SE field. In contrast to previous studies, they conducted a keyword search to identify the articles; hence, they did not include all articles in specific publications. Similarly, as Garousi and Mäntylä [3] did not report the full list of conferences and journals considered in their dataset, neither of the data sources of these studies could be compared with our results.

From this perspective, it can be concluded that the majority of the corpus of earlier studies [6], [8]–[60] were included in the current study. However, it should also be noted that in the field of SE, there is no consensus nor standardized objective methodological approach for creating a corpus. Topic models are defined as statistical models for discovering the latent topics of a corpus of text documents [61]; therefore, creating the corpus for the text documents is important as it constitutes a critical step for this type of analysis and significantly affects the results. Therefore, the current study proposes a very systematic and objective methodology for defining the dataset in order to better understand the topics and trends of SE. The same systematic approach proposed in the current study can easily be used to compare the results of upcoming studies to obtain a better trend estimation for the future of the field.

B. SE CLASSIFICATION SYSTEMS

Cosh *et al.* [7] identified the themes and trends in SE using hierarchical clustering. They analyzed 8,845 articles published between 1995 and 2013 and identified the 20 largest SE clusters. However, even though some common topics were found in their study ("Project", "Software Metrics", "Process Modeling", "Testing", "Requirements", and "Web Services"), some topics that were identified in the current study ("Empirical Software Engineering", "Architecture", "Logic Programming", "Programming Languages", "Parallel Computing", "Specification and Verification", "Source Code", "Program Analysis", "Algorithm", "Object-Oriented", "Constraint Optimization", "Data", "Distributed Systems", "Concurrency", "Mobile", and "Security") were not directly addressed by Cosh *et al.* [7]. Additionally, the topics "Component/Components/Interfaces", "Agent/Agents/Mas", "Software/Engineering/Maintenance", "Fuzzy/Membership/Operator", "Quality/Iso/Software", "Reliability/Failure/Software", "Simulation/Model/Fms", "Knowledge/Km/Kbs", and "Students/Course/Student" that were determined as the largest clusters of SE by Cosh *et al.* [7] were not directly identified in our study. The time interval differences between these two studies could have been the cause of the dissimilarities, i.e., the Cosh *et al.* [7] study covered 18 years, whereas the current study investigated 40 years of publications, from 1980 to 2019.

TABLE 4. Common topics in the current study and previous studies.

Current Study	Cosh *et al.* [7]	Mathew *et al.* [8]	Barua *et al.* [9]
Empirical Software Engineering			
Project	✓		✓
Architecture		✓	
Logic Programming			
Software Metrics	✓	✓	
Programming Languages			
Parallel Computing			
Specification and Verification			
Source Code		✓	✓
Program Analysis		✓	
Process Modeling	✓	✓	
Design Tools			
Algorithm			✓
Testing	✓	✓	✓
Requirements	✓		
Object-Oriented			✓
Constraint Optimization			
Data			✓
Distributed Systems			
Web Services	✓		✓
Fault Detection			
Concurrency			
Mobile			✓
Security			✓

Garousi and Mäntylä [3] identified 67 topics by analyzing 70,000 articles published between 1968 and 2014. However, as they did not provide a full list of these topics, it is difficult to compare their topics with those found in the current study. Mathew *et al.* [8] applied text-mining techniques to 35,391 SE research articles and identified ten major topics, of which six are common to the current study (''Architecture'', ''Metrics'', ''Source Code'', ''Program Analysis'', ''Software Process'', and ''Testing''), whereas the four topics that were not included in the results of the current study were ''Applications'', ''Performance'', ''Modeling'', and ''Developer''. However, Mathew *et al.* also reported that the topic ''Performance'' was far less popular than many others, and that ''Modeling'' occurred with a decreasing frequency in their sample [8]. Because the study conducted by Mathew *et al.* [8] covered a time period of 24 years, it is natural that differences would occur between the current study and theirs. Barua *et al.* [9] also attempted to identify SE topics and trends by analyzing the Stack Overflow database according to the communication of software developers. Their scope and dataset were different from the current study; however, there were some common topics (''Project'', ''Source Code'', ''Algorithm'', ''Testing'', ''Object-Oriented'', ''Data'', ''Web Services'', ''Mobile'', and ''Security''). Although these studies differ in scope and method, they basically support each other (Table 4) by validating the identified SE topics. However, the above-mentioned differences were observed among the emerging topics due to the fact that our study was conducted using a more systematic approach as well as having a wider scope that included recent publications.

TABLE 5. Topics identified and the ACM SE classification system.

The ACM Computing Classification System D.2 SOFTWARE ENGINEERING	Similar Topics Identified
D.2.1 Requirements/Specifications	Requirements
D.2.2 Design Tools and Techniques	Design Tools
D.2.3 Coding Tools and Techniques	Source Code
D.2.4 Software/Program Verification	Specification and Verification
D.2.5 Testing and Debugging	Testing
D.2.6 Programming Environments	Programming Languages
D.2.7 Distribution, Maintenance, and Enhancement	Distributed Systems
D.2.8 Metrics	Software Metrics
D.2.9 Management	Project
D.2.10 Design	
D.2.11 Software Architectures	Architecture
D.2.12 Interoperability	
D.2.13 Reusable software	

In 1998, the Association for Computing Machinery (ACM) published a computing classification system for software engineering (D.2) [62], which is very helpful for establishing the main SE system components. Table 5 shows the identified topics compared to the ACM classifications [62].

Table 5 shows that ''D.2.9 Management'' can be mapped to the topic ''Project'', as it also covers management

problems (Table 3). Furthermore, except for ''D.2.13 Reusable software'', ''D.2.12 Interoperability'', and ''D.2.10 Design'', all the ACM classifications can be mapped to the identified topics.

In 2006, Cai and Card [6] conducted a review study to analyze SE research topics, in which they mapped all research studies according to the ACM classification system. Their results indicated that the topics D.2.13, D.2.12, and D.2.10 were those with the lowest ranking [6]; i.e., they revealed that studies addressing these three topics were ranked as the lowest. According to our results, the software design items were classified under the topic ''Design Tools'', and similarly, the ''reusable software'' keyword was classified under the topic ''Architecture''. The results of the current study indicated the importance of 14 topics which are not classified under the ACM schema (''Empirical Software Engineering'', ''Logic Programming'', ''Parallel Computing'', ''Program Analysis'', ''Process Modeling'', ''Algorithm'', ''Object-Oriented'', ''Constraint Optimization'', ''Database'', ''Web Services'', ''Fault Detection'', ''Concurrency'', ''Mobile'', and ''Security'').

In conclusion, a more standard classification approach can be provided for the SE community by standardizing the corpus creation mechanisms and analysis methods. The classifications may also be updated in a regular and more systematic manner by regularly applying these techniques to the SE studies literature.

C. DEVELOPMENTAL STAGES OF SE

In the analysis of the developmental stages of SE, our results indicated that during the years between 2000 and 2005, one of the top topics was ''Object-Oriented'', and between 2005 and 2010, ''Process Modeling'' (Fig. 7). This finding supports the results reported by Hamadicharef [61], showing a keyword cloud highlighted by ''Object'' from 2000 to 2005, and ''Model'' from 2005 to 2009.

The developmental stage of the SE field between 1980 and 1995 is referred to as the ''Programming Age'' in the current study. As reported by Boehm [63], in that time period, SE studies were following a highly formal and mathematical approach and did not take responsibility for developing system requirements. Hence, they followed minimal reductionist software development practices by separating many essential stages of the software development process. The study by Garousi and Mäntylä [3] revealed that the topic ''Programming'' was dominant during the 1970s and 1980s; furthermore, in the 1990s, it was still one of the dominating topics. These earlier findings, in parallel with the current study, referred to the earlier period of SE from 1980 to 1995 as the ''programming age''.

As reported in earlier studies, due to poorly defined requirements, software had become harder to produce, and alternative SE processes were beginning to be developed by involving systems engineering activities [63]. The study of Garousi and Mäntylä [3] demonstrated that ''Development'' was one of the dominating topics during the 2000s. In parallel with these earlier reports, the current study also indicated this focus by recognizing the ''Software Development Age'' between 1995 and 2010, which concentrated on software developmental stages and components.

The developmental stage of the SE field between 2010 and 2020 is referred to as the ''Software Optimization Age''. The study of Garousi and Mäntylä [3] reported that the topics ''Analysis'' and ''Testing'' were among the dominating topics during this time. By presenting a systematic methodology and by using text-mining algorithms (e.g., LDA), the current study offers the novel contribution of specifying the developmental stages of the field. Fig. 7 enables a better understanding of the evolution of the field and offers some clues about its further development.

D. TOPIC TRENDS

In this study, the topics ''Data'', ''Source Code'', and ''Mobile'' exhibited higher recent accelerations (Fig. 5). It is notable that the topic ''Data'' had a moderate overall acceleration value of 0.06 (Fig. 3), whereas it had the highest recent acceleration value of 0.70 (Fig. 5). A similarly interesting result was observed for the topic ''Security'', which displayed a moderate overall acceleration of 0.04 (Fig. 3), but a higher recent acceleration of 0.22 (Fig. 5). These results indicated that these topics had recently accelerated faster. Buyya *et al.* [64] also reported that, with the emergence of cloud computing technologies, new challenges have attracted the attention of researchers and the SE industry in terms of security and new data-modeling approaches. Hence, our results supporting the findings of these earlier studies underline the importance of the ''Security'' and ''Data'' topics for the next decade of SE. This is also an indicator that these topics will possibly dominate the research in the following decades. Additionally, higher overall (0.21 and 0.12, respectively) (Fig. 3) and recent (0.54 and 0.53, respectively) (Fig. 5) acceleration values were found for the topics ''Source Code'' and ''Mobile''. In agreement with these results, Garousi and Mäntylä [3] also reported ''Source Code'' and ''Mobile and Cloud Computing'' as hot topics in the SE field. Similarly, Mathew *et al.* [8] determined that ''Source Code'' had become a very popular topic over the last decade.

Results of this study indicated a moderate overall acceleration for the topics ''Testing'' (0.07) and ''Program Analysis'' (0.01) (Fig. 3). Concurring with these results, Mathew *et al.* [8] also reported that these topics had become very popular over the last decade. Additionally, Cai and Card [6] observed that ''D.2.5 Testing and Debugging'' had rated the highest in popularity. However, our results showed that the acceleration values of ''Testing'' (-0.10) and ''Program Analysis'' (-0.31) had recently decreased (Fig. 6) and thus, these topics are not expected to dominate SE studies in the following decade.

According to the results of the current study, the topic ''Programming Languages'' had a higher volume in the early days of the SE field (Appendix-A, 20.30 T% from 1980 to 1984). However, its volume decreased in the last period

TABLE 6. Trend-lines of each topic identified.

Topic Name		1980-84	1985-89	1990-94	1995-99	2000-04	2005-09	2010-14	2015-19	AVG
Empirical Software Engineering	C%	0.13	0.24	0.38	0.60	0.81	1.21	1.33	1.92	0.83
	Y%	4.91	5.35	5.98	6.99	7.57	6.47	6.14	7.01	6.30
	AC	0.16	0.39	-0.24	0.76	-0.63	0.02	-0.21	0.05	0.04
Project	C%	0.09	0.18	0.35	0.59	0.73	1.24	1.44	1.80	0.80
	Y%	3.15	3.99	5.47	6.91	6.85	6.67	6.72	6.58	5.79
	AC	1.22	-0.24	0.29	0.13	-0.12	0.05	-0.10	-0.18	0.13
Architecture	C%	0.12	0.24	0.38	0.55	0.83	1.10	1.19	1.31	0.72
	Y%	4.59	5.47	6.00	6.51	7.73	6.00	5.55	4.79	5.83
	AC	1.05	-0.38	0.20	0.23	0.29	-0.79	0.08	-0.10	0.07
Logic Programming	C%	0.26	0.40	0.53	0.65	0.60	0.89	1.08	1.03	0.68
	Y%	9.83	8.98	8.36	7.69	5.60	4.93	5.00	3.77	6.77
	AC	-2.00	0.73	0.13	-0.69	-0.43	0.13	0.10	-0.33	-0.30
Software Metrics	C%	0.10	0.19	0.42	0.50	0.54	0.88	1.03	1.33	0.62
	Y%	3.63	4.19	6.68	5.89	5.06	4.75	4.79	4.85	4.98
	AC	0.53	-0.01	0.09	-0.19	-0.06	0.10	-0.11	0.16	0.06
Programming Languages	C%	0.53	0.60	0.49	0.48	0.45	0.68	0.78	0.77	0.60
	Y%	20.30	13.55	7.91	5.67	4.14	3.67	3.62	2.80	7.71
	AC	-1.15	-1.65	-0.80	-0.38	-0.06	-0.08	0.00	-0.22	-0.54
Parallel Computing	C%	0.11	0.26	0.34	0.42	0.51	0.76	1.06	1.28	0.59
	Y%	4.21	5.73	5.47	5.02	4.72	4.19	4.91	4.68	4.87
	AC	-0.09	0.40	-0.24	0.02	-0.13	-0.08	0.28	-0.02	0.02
Specification and Verification	C%	0.11	0.18	0.35	0.48	0.63	0.87	0.98	1.06	0.58
	Y%	4.25	4.14	5.52	5.66	5.90	4.78	4.54	3.89	4.84
	AC	-0.11	0.16	0.58	-0.31	-0.07	-0.06	0.02	-0.26	-0.01
Source Code	C%	0.02	0.04	0.07	0.13	0.20	0.79	1.23	2.18	0.58
	Y%	0.86	0.97	1.15	1.49	1.88	4.20	5.72	7.94	3.03
	AC	0.15	-0.05	0.05	0.15	0.17	0.46	0.23	0.54	0.21
Program Analysis	C%	0.10	0.15	0.28	0.40	0.44	0.78	1.01	1.14	0.54
	Y%	3.78	3.26	4.38	4.70	4.09	4.26	4.69	4.19	4.17
	AC	0.12	-0.18	0.53	-0.29	0.04	0.00	0.16	-0.31	0.01
Process Modeling	C%	0.06	0.11	0.21	0.29	0.49	0.94	1.06	1.13	0.54
	Y%	2.12	2.51	3.32	3.36	4.58	5.11	4.92	4.14	3.76
	AC	0.40	-0.03	0.04	0.14	0.27	0.01	-0.06	-0.09	0.09
Design Tools	C%	0.18	0.36	0.44	0.46	0.52	0.77	0.75	0.82	0.54
	Y%	6.66	8.05	6.97	5.41	4.83	4.12	3.46	3.00	5.31
	AC	0.34	0.36	-0.62	-0.21	0.12	-0.18	-0.22	-0.10	-0.06
Algorithm	C%	0.15	0.24	0.27	0.35	0.35	0.73	0.89	1.19	0.52
	Y%	5.92	5.46	4.23	4.17	3.24	4.06	4.11	4.34	4.44
	AC	-1.16	0.83	-0.31	-0.16	-0.59	0.57	0.07	0.04	-0.09
Testing	C%	0.04	0.06	0.19	0.24	0.34	0.69	1.14	1.41	0.51
	Y%	1.62	1.38	2.94	2.82	3.18	3.70	5.26	5.14	3.26
	AC	-0.13	-0.01	0.46	-0.28	0.24	0.19	0.20	-0.10	0.07
Requirements	C%	0.05	0.08	0.17	0.27	0.41	0.77	0.92	1.18	0.48
	Y%	1.69	1.79	2.74	3.18	3.83	4.15	4.28	4.32	3.25
	AC	0.09	0.06	0.19	0.06	0.23	-0.06	-0.07	0.06	0.07
Object-Oriented	C%	0.08	0.20	0.39	0.49	0.65	0.88	0.64	0.52	0.48
	Y%	2.79	4.42	6.13	5.80	6.01	4.82	2.97	1.92	4.36
	AC	0.33	0.07	0.32	0.17	-0.10	-0.34	-0.31	-0.19	-0.01
Constraint Optimization	C%	0.06	0.11	0.15	0.42	0.46	0.82	0.87	0.89	0.47
	Y%	2.29	2.47	2.36	5.02	4.24	4.58	4.02	3.24	3.53
	AC	-0.12	0.25	-0.04	0.38	-0.38	0.32	-0.18	0.01	0.03
Data	C%	0.11	0.18	0.19	0.21	0.29	0.52	0.67	1.39	0.45
	Y%	4.10	4.04	3.10	2.51	2.70	2.86	3.10	5.08	3.44
	AC	-0.02	-0.17	-0.17	0.08	-0.01	-0.01	0.05	0.70	0.06
Distributed Systems	C%	0.17	0.33	0.31	0.36	0.40	0.54	0.52	0.63	0.41
	Y%	6.26	7.48	4.84	4.24	3.72	3.00	2.41	2.29	4.28
	AC	0.19	-0.23	-0.22	-0.02	-0.21	-0.18	-0.11	0.05	-0.09
Web Services	C%	0.01	0.02	0.03	0.10	0.33	1.01	0.68	0.92	0.39
	Y%	0.46	0.39	0.46	1.10	3.02	5.51	3.15	3.37	2.18
	AC	0.05	-0.06	0.01	0.20	1.14	-0.28	-0.23	-0.31	0.07
Fault Detection	C%	0.05	0.08	0.12	0.16	0.23	0.45	0.77	1.10	0.37
	Y%	1.82	1.71	1.97	1.86	2.18	2.41	3.55	4.02	2.44
	AC	0.08	-0.12	0.08	-0.03	0.15	0.10	0.22	0.04	0.07
Concurrency	C%	0.08	0.15	0.17	0.18	0.23	0.43	0.60	0.59	0.30
	Y%	3.08	3.33	2.74	2.15	2.10	2.33	2.81	2.15	2.59
	AC	0.12	0.01	-0.32	-0.06	0.02	0.14	-0.01	-0.18	-0.04
Mobile	C%	0.02	0.02	0.03	0.08	0.12	0.23	0.43	1.16	0.26

TABLE 6. *(Continued.)* **Trend-lines of each topic identified.**

Topic Name		1980-84	1985-89	1990-94	1995-99	2000-04	2005-09	2010-14	2015-19	AVG
	Y%	0.58	0.43	0.47	0.89	1.14	1.25	1.97	4.23	1.37
	AC	0.09	-0.12	0.06	0.14	-0.01	-0.02	0.25	0.53	0.12
	C%	0.03	0.04	0.05	0.08	0.18	0.39	0.49	0.61	0.23
Security	Y%	1.11	0.91	0.81	0.95	1.68	2.16	2.30	2.24	1.52
	AC	-0.12	-0.01	-0.06	0.16	0.14	-0.01	-0.04	0.22	0.04

C%: Percentage of the topics in the corpus
Y%: Percentage of the topics in the same yearly period
AC: Average acceleration in five-year periods

(Appendix-A, 2.80 T% from 2015 to 2019). A similar trend was seen for the topic ''Logic Programming'' (Appendix-A, 9.83 T% from 1980 to 1984 and 3.77 T% from 2015 to 2019). Hence, with the emergence of new topics such as ''Mobile'' and ''Security'', these topics with slower acceleration can be considered as older topics in the field.

As seen in Fig. 7, starting with the increased focus on programming, the evolution of SE has continued through the concepts of software development and software optimization. Moreover, it can be expected that, through better organization, analysis, and understanding of the data, the future focus of the field will continue to be on the creation of more data-oriented software systems.

VI. CONCLUSION

This study provides four major contributions to the field of SE. First, it proposes a systematic methodology for corpus creation by considering the major studies in the field. The proposed methodology can be applied to topic-modeling analysis in any field in which there are currently not many studies adopting such a systematic approach to this type of analysis. Moreover, in order to reveal up-to-date themes and trends in the primary sub-fields of computer science, the methodology of this study can be implemented for operation systems, computer vision, computer networks, distributed computing, computer architecture, natural language processing, cloud computing, and embedded systems. Second, based on the analyses, the developmental stages of the SE field were determined as the ''Programming Age'', ''Software Development Age'', and ''Software Optimization Age''. Third, the recommendations for improving the SE classification system provide a systematic approach to establishing a classification system for SE that can be regularly updated by analyzing its developmental structures. Such classifications can also be applied to any other field to objectively determine the developments in that field and to support research and industrial decision-making processes. We believe that this classification is important for guiding the curriculum for SE-related educational organizations and industrial organizational structures. Finally, the trends of the identified topics offer some important insights concerning their developmental stages and the future of the field. The results indicate that, the topics ''Data'', ''Source Code'', ''Mobile'', and ''Security'' are expected to be highly studied topics over the next decade. As the topic ''Data'' includes some keywords like ''data mining'', ''learning'', and ''clustering'', it can be concluded that its emergence in the field also indicates big-data analysis. As the technological developments move through a mobile environment, in parallel to this, ''big-data analysis security'' and ''mobile technologies'' will also be popular topics in the field. Additionally, because of the complex structure of software systems, ''source-codes'' can be expected to be one of the dominant SE topics.

According to these results, SE education programs can update their course content by explicitly emphasizing these software system developmental stages, which may provide a better map for helping students to locate various SE concepts. Additionally, the SE industry may also utilize these results to better forecast the future of the field and take appropriate actions and decisions.

APPENDIX A

As an example, the formulation for the calculations of the topic ''Empirical Software Engineering'' are given below step-by-step for the first term (1980-1984):

STEP 1. Yearly topic distribution per article was identified by LDA, for each topic found. For the topic ''Empirical Software Engineering'', it was 10.85, 20.52, 13.03, 11.81, and 18.08 for the years 1980 through 1984, respectively.

STEP 2. The sum of yearly distributions (prepared in STEP 1) were calculated for each term. Thus, for the topic ''Empirical Software Engineering'', the sum was calculated as 74.29 for the first term (1980-1984) by adding up the yearly distributions of 10.85, 20.52, 13.03, 11.81, and 18.08 for each year of the first term.

STEP 3. The C% (percentage of the topics in the corpus) values were calculated through a normalized score. This score was calculated for each term for each topic by taking the sum of yearly distributions for each term (as calculated in STEP 2) to all the articles in the corpus (57,174 for this study). For example, for the topic ''Empirical Software Engineering'' for the first term (1980-1984), the C% was calculated as 0.13 (74.29 * 100/57,174).

STEP 4. The sum of yearly distributions for each year for all topics were calculated. For example, for the year 1980, it was 10.85 for the topic ''Empirical Software Engineering'', 3.07 for the topic ''Project'', and 6.29 for the topic ''Architecture''. At the end, for the year 1980, the sum of all yearly distributions for all topics was found as 229.

STEP 5. A normalized score (NS) for each year and each topic was calculated by taking the percentage of the yearly distributions of a specific topic for each year to the total number of articles for all topics in the same year (as calculated in Step 4); e.g., for the topic ''Empirical Software Engineering'', this percentage was calculated as $10.85 * 100/229 = 4.74$ for the year 1980. Similar calculations were performed for the same topic as 6.66, 3.78, 3.99, and 5.38 for the years 1981, 1982, 1983, and 1984, respectively.

STEP 6. The Y% (percentage of the topics in the same yearly period) value was calculated by taking the average of NSs in STEP 5, for each topic and term; e.g., for the topic ''Empirical Software Engineering'', the Y% was calculated as 4.91 for the term 1980-1984 by taking the average of values 4.74, 6.66, 3.78, 3.99, and 5.38.

STEP 7. The acceleration values were calculated for each year by subtracting the NSs of previous years (as calculated in Step 6) from that of the current year. For the first term, the acceleration value for the year 1980 was not calculated because the acceleration value for year 1979 was unknown. Other than this, an acceleration value was calculated for each year. For the topic ''Empirical Software Engineering'', it was found as 1.93, −2.89, 0.21, and 1.39 for the years 1981, 1982, 1983, and 1984, respectively.

STEP 8. The average of these acceleration values was calculated and the result recorded as the AC (Average acceleration of five-year periods) value for the related term of the topic, e.g., for the topic ''Empirical Software Engineering'', by finding the average of the acceleration values (1.93, −2.89, 0.21, 1.39), the AC value for the first term (1980-1984) was calculated as 0.16.

REFERENCES

[1] B. Kitchenham and P. Brereton, ''A systematic review of systematic review process research in software engineering,'' *Inf. Softw. Technol.*, vol. 55, no. 12, pp. 2049–2075, Dec. 2013.

[2] S. MacDonell, M. Shepperd, B. Kitchenham, and E. Mendes, ''How reliable are systematic reviews in empirical software engineering?'' *IEEE Trans. Softw. Eng.*, vol. 36, no. 5, pp. 676–687, Sep. 2010.

[3] V. Garousi and M. V. Mäntylä, ''Citations, research topics and active countries in software engineering: A bibliometrics study,'' *Comput. Sci. Rev.*, vol. 19, pp. 56–77, Feb. 2016.

[4] P. Brereton, B. A. Kitchenham, D. Budgen, M. Turner, and M. Khalil, ''Lessons from applying the systematic literature review process within the software engineering domain,'' *J. Syst. Softw.*, vol. 80, no. 4, pp. 571–583, Apr. 2007.

[5] T. Hall, N. Baddoo, S. Beecham, H. Robinson, and H. Sharp, ''A systematic review of theory use in studies investigating the motivations of software engineers,'' *ACM Trans. Softw. Eng. Methodol.*, vol. 18, no. 3, pp. 1–29, May 2009.

[6] K. Y. Cai and D. Card, ''An analysis of research topics in software engineering–2006,'' *J. Syst. Softw.*, vol. 81, no. 6, pp. 1051–1058, 2008.

[7] K. Cosh, S. Ramingwong, N. Eiamkanitchat, and L. Ramingwong, ''Automatically identifying themes and trends in software engineering research,'' in *Proc. 10th Int. Conf. Knowl. Smart Technol. (KST)*, Jan. 2018, pp. 106–111.

[8] G. Mathew, A. Agrawal, and T. Menzies, ''Finding trends in software research,'' *IEEE Trans. Softw. Eng.*, early access, Sep. 14, 2018, doi: 10.1109/TSE.2018.2870388.

[9] A. Barua, S. W. Thomas, and A. E. Hassan, ''What are developers talking about? An analysis of topics and trends in stack overflow,'' *Empirical Softw. Eng.*, vol. 19, no. 3, pp. 619–654, Jun. 2014.

[10] C. Wohlin, ''An analysis of the most cited articles in software engineering journals–2000,'' *Inf. Softw. Technol.*, vol. 49, no. 1, pp. 2–11, Jan. 2007.

[11] C. Wohlin, ''An analysis of the most cited articles in software engineering journals–1999,'' *Inf. Softw. Technol.*, vol. 47, no. 15, pp. 957–964, 2005.

[12] C. Wohlin, ''An analysis of the most cited articles in software engineering journals–2001,'' *Inf. Softw. Technol.*, vol. 50, nos. 1–2, pp. 3–9, 2008.

[13] B. Kitchenham, ''What's up with software metrics?–A preliminary mapping study,'' *J. Syst. Softw.*, vol. 83, no. 1, pp. 37–51, 2010.

[14] V. Garousi and J. M. Fernandes, ''Highly-cited papers in software engineering: The top-100,'' *Inf. Softw. Technol.*, vol. 71, pp. 108–128, Mar. 2016.

[15] W. E. Wong, T. H. Tse, R. L. Glass, V. R. Basili, and T. Y. Chen, ''An assessment of systems and software engineering scholars and institutions (2002–2006),'' *J. Syst. Softw.*, vol. 82, no. 8, pp. 1370–1373, 2009.

[16] W. E. Wong, T. H. Tse, R. L. Glass, V. R. Basili, and T. Y. Chen, ''An assessment of systems and software engineering scholars and institutions (2003–2007 and 2004–2008),'' *J. Syst. Softw.*, vol. 84, no. 1, pp. 162–168, 2011.

[17] W. E. Wong, T. H. Tse, R. L. Glass, V. R. Basili, and T. Y. Chen, ''An assessment of systems and software engineering scholars and institutions (2001–2005),'' *J. Syst. Softw.*, vol. 81, no. 6, pp. 1059–1062, 2008.

[18] V. Garousi and T. Varma, ''A bibliometric assessment of Canadian software engineering scholars and institutions (1996–2006),'' *Comput. Inf. Sci.*, vol. 3, no. 2, pp. 1–11, Apr. 2010.

[19] F. G. De Freitas and J. T. De Souza, ''Ten years of search based software engineering: A bibliometric analysis,'' in *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Berlin, Germany: Springer, 2011.

[20] R. Farhoodi, V. Garousi, D. Pfahl, and J. Sillito, ''Development of scientific software: A systematic mapping, a bibliometrics study, and a paper repository,'' *Int. J. Softw. Eng. Knowl. Eng.*, vol. 23, no. 4, pp. 463–506, May 2013.

[21] V. Garousi and G. Ruhe, ''A bibliometric/geographic assessment of 40 years of software engineering research (1969–2009),'' *Int. J. Softw. Eng. Knowl. Eng.*, vol. 23, no. 9, pp. 1343–1366, 2013.

[22] V. Garousi, ''A bibliometric analysis of the Turkish software engineering research community,'' *Scientometrics*, vol. 105, no. 1, pp. 23–49, Oct. 2015.

[23] J. M. Fernandes, ''Authorship trends in software engineering,'' *Scientometrics*, vol. 101, no. 1, pp. 257–271, Oct. 2014.

[24] V. Garousi, K. Petersen, and B. Ozkan, ''Challenges and best practices in industry-academia collaborations in software engineering: A systematic literature review,'' *Inf. Softw. Technol.*, vol. 79, pp. 106–127, Nov. 2016.

[25] V. Garousi and M. V. Mäntylä, ''A systematic literature review of literature reviews in software testing,'' *Inf. Softw. Technol.*, vol. 80, pp. 195–216, Dec. 2016.

[26] A. Ramirez, J. R. Romero, and C. L. Simons, ''A systematic review of interaction in search-based software engineering,'' *IEEE Trans. Softw. Eng.*, vol. 45, no. 8, pp. 760–781, Aug. 2019.

[27] S. Jalali and C. Wohlin, ''Global software engineering and agile practices: A systematic review,'' *J. Software: Evol. Process*, vol. 24, no. 6, pp. 643–659, Oct. 2012.

[28] B. Kitchenham, ''Procedures for undertaking systematic reviews'' Dept. Comput. Sci., Keele Univ., Nat. ICT Aust. Ltd., Joint Tech. Rep. TR/SE-0401 and 0400011T.1, 2004.

[29] J. Biolchini, P. G. Mian, A. C. C. Natali, and G. H. Travassos, ''Systematic review in software engineering,'' Univ. Federal do Rio de Janeiro, Brazil, Tech. Rep. ES 679-5, 2005, p. 45.

[30] S. Kitchenham and B. Charters, ''Guidelines for performing systematic literature reviews in software engineering,'' Keele Univ., Durham Univ. Joint Rep., U.K., Tech. Rep. EBSE 2007-001, Ver. 2.3, 2007.

[31] B. A. Kitchenham, S. L. Pfleeger, and L. M. Pickard, ''Preliminary guidelines for empirical research in software engineering,'' *IEEE Trans. Softw. Eng.*, vol. 28, no. 8, pp. 721–734, Aug. 2002.

[32] F. Q. B. da Silva, A. L. M. Santos, S. Soares, A. C. C. França, C. V. F. Monteiro, and F. F. Maciel, ''Six years of systematic literature reviews in software engineering: An updated tertiary study,'' *Inf. Softw. Technol.*, vol. 53, no. 9, pp. 899–913, Sep. 2011.

[33] B. Kitchenham, O. Pearl Brereton, D. Budgen, M. Turner, J. Bailey, and S. Linkman, ''Systematic literature reviews in software engineering—A systematic literature review,'' *Inf. Softw. Technol.*, vol. 51, no. 1, pp. 7–15, Jan. 2009.

[34] J. M. Verner, O. P. Brereton, B. A. Kitchenham, M. Turner, and M. Niazi, ''Systematic literature reviews in global software development: A tertiary study,'' in *IET Seminar Dig.*, May 2012, pp. 2–11.

[35] Y. Shakeel, J. Krüger, I. Von Nostitz-Wallwitz, G. Saake, and T. Leich, "Automated selection and quality assessment of primary studies: A systematic literature review," *J. Data Inf. Qual.*, vol. 12, no. 1, pp. 1–26, 2019.
[36] G. Rodríguez-Pérez, R. Nadri, and M. Nagappan, "Perceived diversity in software engineering: A systematic literature review," *Empirical Softw. Eng.*, vol. 26, no. 5, pp. 1–38, Sep. 2021.
[37] L. Zhang, J. H. Tian, and J. Jiang, "Empirical research in software engineering-A literature survey," *J. Comput. Sci. Technol.*, vol. 33, no. 5, pp. 876–899, 2018.
[38] N. Nazar, Y. Hu, and H. Jiang, "Summarizing software artifacts: A literature review," *J. Comput. Sci. Technol.*, vol. 31, pp. 883–909, Sep. 2016.
[39] K. R. Felizardo, N. Salleh, R. M. Martins, E. Mendes, S. G. Macdonell, and J. C. Maldonado, "Using visual text mining to support the study selection activity in systematic literature reviews," in *Proc. Int. Symp. Empirical Softw. Eng. Meas.*, Sep. 2011, pp. 77–86.
[40] E. Hassler, J. C. Carver, D. Hale, and A. Al-Zubidy, "Identification of SLR tool needs–results of a community workshop," *Inf. Softw. Technol.*, vo. 70, pp. 122–129, Feb. 2016.
[41] X. Sun, X. Liu, B. Li, Y. Duan, H. Yang, and J. Hu, "Exploring topic models in software engineering data analysis: A survey," in *Proc. 17th IEEE/ACIS Int. Conf. Softw. Eng., Artif. Intell., Netw. Parallel/Distrib. Comput. (SNPD)*, May 2016, pp. 357–362.
[42] D. Binkley, D. Heinz, D. Lawrie, and J. Overfelt, "Source code analysis with LDA," *J. Software: Evol. Process*, vol. 28, no. 10, pp. 893–920, Oct. 2016.
[43] F. Gurcan and N. E. Cagiltay, "Big data software engineering: Analysis of knowledge domains and skill sets using LDA-based topic modeling," *IEEE Access*, vol. 7, pp. 82541–82552, 2019.
[44] F. Gurcan, N. E. Cagiltay, and K. Cagiltay, "Mapping human–computer interaction research themes and trends from its existence to today: A topic modeling-based review of past 60 years," *Int. J. Hum. Comput. Interact.*, vol. 37, no. 3, pp. 267–280, 2021.
[45] F. Gurcan and N. E. Cagiltay, "Research trends on distance learning: A text mining-based literature review from 2008 to 2018," *Interact. Learn. Environments*, to be published.
[46] P. Mongeon and A. Paul-Hus, "The journal coverage of web of science and scopus: A comparative analysis," *Scientometrics*, vol. 106, no. 1, pp. 213–228, Jan. 2016.
[47] M. Konrad. (2017). *Text Mining and Topic Modeling Toolkit*. Accessed: Jan. 21, 2022. [Online]. Available: https://pypi.org/project/tmtoolkit/
[48] F. Gurcan and N. E. Cagiltay, "Exploratory analysis of topic interests and their evolution in bioinformatics research using semantic text mining and probabilistic topic modeling," *IEEE Access*, vol. 10, pp. 31480–31493, 2022.
[49] M. Porter, "Snowball: A language for stemming algorithms," *Snowball*, to be published.
[50] A. K. Uysal and S. Gunal, "The impact of preprocessing on text classification," *Inf. Process. Manage.*, vol. 50, no. 1, pp. 104–112, Jan. 2014.
[51] D. M. Blei and J. D. Lafferty, "Correction: A correlated topic model of science," *Ann. Appl. Statist.*, vol. 1, no. 2, p. 634, Dec. 2007.
[52] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent Dirichlet allocation," *J. Mach. Learn. Res.*, vol. 3, nos. 4–5, pp. 993–1022, 2003.
[53] I. Vayansky and S. A. P. Kumar, "A review of topic modeling methods," *Inf. Syst.*, vol. 94, Dec. 2020, Art. no. 101582.
[54] *Tomotopy Topic Modeling Package*. Accessed: Feb. 27, 2022. [Online]. Available: https://bab2min.github.io/tomotopy/v0.12.2/en/
[55] D. M. Blei, "Probabilistic topic models," *Commun. ACM*, vol. 55, no. 4, pp. 77–84, Apr. 2012.
[56] F. Gurcan, O. Ozyurt, and N. E. Cagitay, "Investigation of emerging trends in the E-learning field using latent Dirichlet allocation," *Int. Rev. Res. Open Distrib. Learn.*, vol. 22, no. 2, pp. 1–18, Jan. 2021.
[57] F. Gurcan, "Extraction of core competencies for big data: Implications for competency-based engineering education," *Int. J. Eng. Educ.*, vol. 35, no. 4, pp. 1110–1115, 2019.
[58] Y. Zuo, J. Wu, H. Zhang, H. Lin, F. Wang, and K. Xu, "Topic modeling of short texts: A pseudo-document view," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2016, pp. 2105–2114.
[59] D. Mimno, H. M. Wallach, E. Talley, M. Leenders, and A. McCallum, "Optimizing semantic coherence in topic models," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2011, pp. 1–11.
[60] S. Datta, S. Sarkar, and A. S. M. Sajeev, "How long will this live? Discovering the lifespans of software engineering ideas," *IEEE Trans. Big Data*, vol. 2, no. 2, pp. 124–137, Jun. 2016.
[61] B. Hamadicharef, "Scientometric study of the IEEE transactions on software engineering 1980–2010," in *Proc. 2nd Int. Congr. Comput. Appl. Comput. Sci.* (Advances in Intelligent and Soft Computing). Berlin, Germany: Springer, 2012, pp. 101–106.
[62] *The ACM Computing Classification System*, 1998.
[63] B. Boehm, "Some future trends and implications for systems and software engineering processes," *Syst. Eng.*, vol. 9, no. 1, pp. 1–19, 2006.
[64] R. Buyya, S. N. Srirama, G. Casale, and R. Calheiros, "A manifesto for future generation cloud computing: Research directions for the next decade," *ACM Comput. Surv.*, vol. 51, no. 5, pp. 1–38, 2019.

FATIH GURCAN received the Ph.D. degree in computer engineering from Karadeniz Technical University. He has worked as an Instructor with the Department of Informatics, Karadeniz Technical University, from 2001 to 2014. He has been with the Center for Research and Application in Distance Education, Karadeniz Technical University, since 2015, as an Instructor. His research interests include trend analysis, sentiment analysis, statistical topic modeling, engineering education, data mining, machine learning, big data analytics, and text mining.

GONCA GOKCE MENEKSE DALVEREN received the Ph.D. degree in software engineering from Atilim University. She completed her postdoctoral research at the Department of Computer Science, Norwegian University of Science and Technology. She is currently working at the Software Engineering Department, Atilim University. Her research interests include software engineering, eye tracking, medical informatics, and human–computer interaction.

NERGIZ ERCIL CAGILTAY received the Ph.D. degree in instructional technologies from Middle East Technical University. She has worked for commercial and government organizations as a Project Manager for more than eight years in Turkey. She has also worked for the Indiana University Digital Library Program as a System Analyst and a Programmer for four years. She has been with the Software Engineering Department, Atilim University, Turkey, since 2003, as a Professor. Her main research interests include information systems, medical information systems, engineering education, instructional systems technologies, distance education, e-learning, and medical education.

AHMET SOYLU received the Ph.D. degree in computer science from the University of Leuven, in 2012. He is currently an Associate Professor of computer science at the Norwegian University of Science and Technology and a Professor at Oslo Metropolitan University. His main research interests include ontology-driven information systems and ontology-driven design of information systems from a human–computer interaction perspective.